Serial ATA II:

Port Multiplier

Revision 1.0
25-March-2003

APT Technologies, Inc.
Dell Computer Corporation
Intel Corporation
Maxtor Corporation
Seagate Technology

This 1.0 revision of the Serial ATA II: Port Multiplier specification ("Final Specification") is available for download at www.serialata.org.

For more information about Serial ATA, refer to the Serial ATA working Group website at www.serialata.org.

Serial ATA II Workgroup Port Multiplier Technical Editor:

Amber Huffman
Intel Corporation
2111 NE 25th Ave M/S JF2-53
Hillsboro, OR 97124 USA
Tel: (503) 264-7929
Email: amber.huffman@intel.com

Table of Contents

1. Introduction

A Port Multiplier is a mechanism for one active host connection to communicate with multiple devices. A Port Multiplier can be thought of as a simple multiplexer where one active host connection is multiplexed to multiple device connections, as shown in Figure 1.

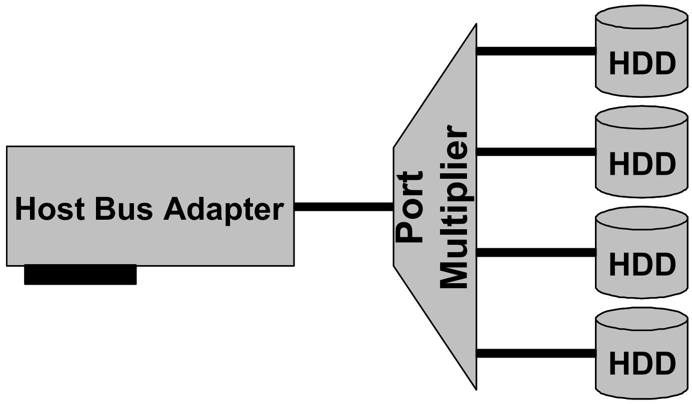

Figure 1 Port Multiplier Overview

Only one active host connection to the Port Multiplier is supported. The Port Multiplier is an extensible design that supports up to 15 device connections and utilizes the full bandwidth of the host connection.

1.1. Goals, Objectives, & Constraints

This specification defines a fan-out device that can be used with Serial ATA 1.0 devices and next generation Serial ATA devices.

Some of the goals and requirements for the Port Multiplier include:

- Serial ATA 1.0 devices may be attached without modification
- Link and Phy layer compatibility with Serial ATA 1.0 must be maintained for both hosts and devices
- No new primitives may be added as part of the definition
- No new FIS types may be added as part of the definition

Some of the constraints include:

- Only one active host connection is supported
- A Port Multiplier shall not be plugged into another Port Multiplier (i.e. no cascading)
- There is a maximum fan-out of 15 device connections

1.2. Example Applications

One possible application of the Port Multiplier is to increase the number of Serial ATA connections in an enclosure that does not have a sufficient number of Serial ATA connections for all of the drives in the enclosure. An example is shown in Figure 2. A multi-lane cable with two Serial ATA connections is delivered to the enclosure. The enclosure contains eight Serial ATA drives. To create the appropriate number of Serial ATA connections, two 1-to-4 Port Multipliers are used to create eight Serial ATA connections.

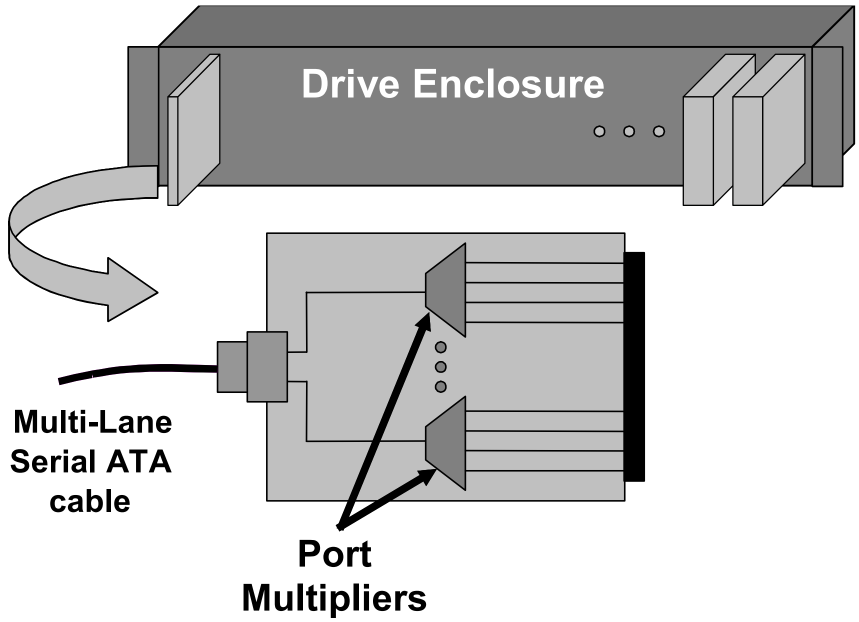

Figure 2 Enclosure example using Port Multipliers with Serial ATA as the connection within the rack

Another example is shown in Figure 3. Fibre Channel, Infiniband, or Gigabit Ethernet is used as the connection within the rack to the enclosure. Inside the enclosure, a host controller creates two Serial ATA connections from the connection delivered. The enclosure contains eight Serial ATA drives. To create the appropriate number of Serial ATA connections, two 1-to-4 Port Multipliers are used to create eight Serial ATA connections.

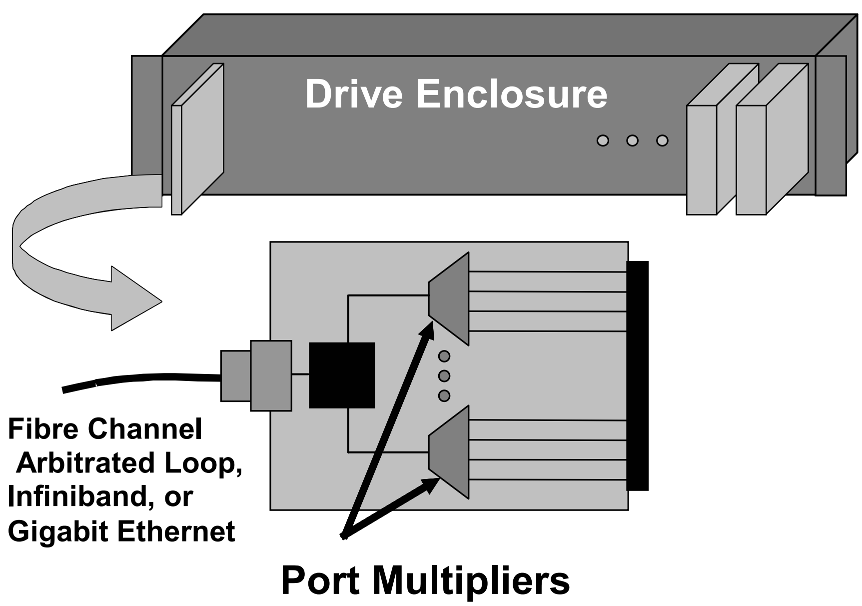

Figure 3 Enclosure example using Port Multipliers with a different connection within the rack

The Port Multiplier allows host controllers with a modest number of connections to be used in these enclosures and then the connectivity is increased as product requirements dictate.

Another example application is using a Port Multiplier to increase the number of Serial ATA connections in a mobile docking station. The example shown in Figure 4 has a proprietary interface between the laptop and the docking station. The proprietary interface may route a Serial ATA connection from the laptop to the docking station or the docking station may create a Serial ATA connection itself. The docking station routes the Serial ATA connection to a Port Multiplier to create an appropriate number of Serial ATA connections for the number of devices to be attached.

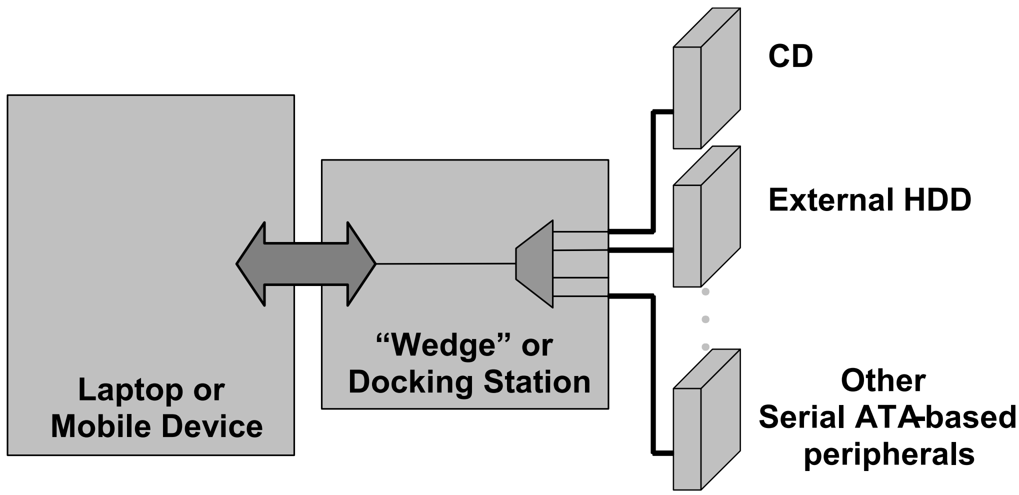

Figure 4 Mobile docking station example using a Port Multiplier

These are a few examples of possible applications of the Port Multiplier and are not meant to be all encompassing.

1.3. Definitions, abbreviations, and conventions

1.3.1. Definitions and Abbreviations

The terminology used in this specification is consistent with the terminology used in the Serial ATA 1.0 and Serial ATA II specifications, and all definitions and abbreviations defined in those specifications are used consistently in this document. Additional terms and abbreviations introduced in this specification are defined in the following sections.

1.3.1.1. Control port

The Port Multiplier has one port address reserved for control and status communication with the Port Multiplier itself. The control port has port address Fh.

1.3.1.2. Device port

A device port is a port that can be used to connect a device to the Port Multiplier. The Port Multiplier may support up to 15 device ports. The device port addresses shall start at zero and shall be numbered sequentially until all device ports have port addresses. A valid device port shall have a port address that has a value less than the total number of device ports supported by the Port Multiplier.

1.3.1.3. Host port

The host port is the port that is used to connect the Port Multiplier to the host. There is only one active host port.

1.3.1.4. Port address

The control port and each device port present on a Port Multiplier have a port address. The port address is used to route FISes between the host and a specific device or the control port.

1.4. References

This specification is an extension to the Serial ATA 1.0 specification. The Serial ATA 1.0 specification is presumed as the underlying baseline for this specification. This specification makes reference to the following specifications:

Serial ATA: High Speed Serialized AT Attachment revision 1.0.

Serial ATA II Specification: Extensions to Serial ATA 1.0.

The specifications are available for download at www.serialata.org.

2. Overview

The Port Multiplier uses four bits, known as the PM Port field, in all FIS types to route FISes between the host and the appropriate device. Using the PM Port field, the Port Multiplier can route FISes to up to 15 Serial ATA devices from one active host. The PM Port field is filled in by the host on a host-to-device FIS with the port address of the device to route the FIS to. For a device-to-host FIS, the PM Port field is filled in by the Port Multiplier with the port address of the device that is transmitting the FIS. Device port addresses start at zero and are numbered sequentially higher until the last device port address has been defined. The control port, port address Fh, is used for control and status communication with the Port Multiplier itself.

In order to utilize all devices connected to a Port Multiplier, the host must have a mechanism to set the PM Port field in all transmitted FISes.

The Port Multiplier maintains a set of general purpose registers and also maintains the Serial ATA superset Status and Control registers for each device port. The control port supports two commands, Read Port Multiplier and Write Port Multiplier, that are used to read and write these registers.

Some additional Port Multiplier features include:

- Supports booting with legacy software on device port 0h
- Supports staggered spin-up
- Supports hot plug

3. Definition

3.1. Addressing Mechanism

The Port Multiplier uses four bits, known as the PM Port field, in all FIS types to route FISes between the host and the appropriate device. Using the PM Port field, the Port Multiplier can route FISes to up to 15 Serial ATA devices from one active host. The PM Port field is filled in by the host on a host-to-device FIS with the port address of the device to route the FIS to. For a device-to-host FIS, the PM Port field is filled in by the Port Multiplier with the port address of the device that is transmitting the FIS.

The PM Port field is reserved in the Serial ATA 1.0 definition. Therefore all Serial ATA 1.0 devices shall ignore this field in accordance with Serial ATA 1.0.

3.1.1. FIS Modifications

The PM Port field is in the first Dword of all FIS types. The PM Port field corresponds to reserved bits common to all FIS types in the Serial ATA 1.0 specification. The first Dword of all FIS types is shown in Figure 5.

As defined in Serial ATA 1.0	PM Port	FIS Type

Figure 5 First Dword of all FIS Types

Field Definitions

FIS Type – Defines FIS Type specific fields and FIS length

PM Port – Specifies the port address that the FIS should be delivered to or is received from

3.1.2. Transmission from Host to Device

To transmit a FIS to a device connected to a Port Multiplier, the host shall set the PM Port field in the FIS to the device's port address. Then the host shall start transmitting the FIS to the Port Multiplier in accordance with the Serial ATA 1.0 Transport and Link state machines.

When a Port Multiplier receives a FIS over the host port, the Port Multiplier shall check the PM Port field in the FIS to determine the port address that the FIS should be transmitted over. If the FIS is destined for the control port, the Port Multiplier shall receive the FIS and perform the command or operation requested. If the FIS is destined for a device port, the Port Multiplier shall perform the following procedure:

1. The Port Multiplier shall determine if the device port is valid. If the device port is not valid, the Port Multiplier shall issue a SYNC_P primitive to the host and terminate reception of the FIS. Refer to section 3.2.8.3.

2. The Port Multiplier shall determine if the X bit is set in the device port's PSCR[1] (SError) register. If the X bit is set, the Port Multiplier shall issue a SYNC_P primitive to the host and terminate reception of the FIS. Refer to section 3.2.8.2.

3. The Port Multiplier shall determine if a collision has occurred. A collision is when a reception is already in progress from the device that the host wants to transmit to. If a collision has occurred, the Port Multiplier shall finish receiving the FIS from the host and shall issue an $\mathrm{R\_ERR}_P$ primitive to the host as the ending status. The Port Multiplier shall follow the procedures outlined in section 3.2.2 to clear the collision condition.

4. The Port Multiplier shall initiate the transfer with the device by issuing an $\mathrm{X\_RDY}_P$ primitive to the device. A collision may occur as the Port Multiplier is issuing the $\mathrm{X\_RDY}_P$ to the device if the device has just decided to start a transmission to the host. If the device starts transmitting $\mathrm{X\_RDY}_P$ to the Port Multiplier, a collision has occurred. If a collision has occurred, the Port Multiplier shall finish receiving the FIS from the host and shall issue an $\mathrm{R\_ERR}_P$ primitive to the host as the ending status. The Port Multiplier shall follow the procedures outlined in section 3.2.2 to clear the collision condition.

5. After the device issues $\mathrm{R\_RDY}_P$ to the Port Multiplier, the Port Multiplier shall transmit the FIS from the host to the device. The Port Multiplier shall not send $\mathrm{R\_OK}_P$ status to the host until the device has issued an $\mathrm{R\_OK}_P$ for the FIS reception. The $\mathrm{R\_OK}_P$ status handshake shall be interlocked from the device to the host. Refer to section 3.2.1.

If an error is detected during any part of the FIS transfer, the Port Multiplier shall ensure that the error condition is propagated to the host and the device.

The transfer between the host and Port Multiplier is handled separately from the transfer between the Port Multiplier and device; only the end of frame R_OK_P handshake is interlocked. The Port Multiplier shall ensure that the flow control signaling latency requirement specified in section 7.4.7 of the Serial ATA 1.0 specification is met for all FIS transfers on a per link basis. Specifically, the Port Multiplier shall ensure that the flow control signaling latency is met between:

1. The host port and the host it is connected to
2. Each device port and the device that it is connected to

If there is not an error detected during the FIS transfer, the Port Multiplier shall not alter the FIS transmitted to the device. The Port Multiplier is not required to check or recalculate the CRC.

The Port Multiplier shall have Link and Phy layer state machines that comply with the Serial ATA 1.0 Link and Phy layer state machines.

3.1.3. Transmission from Device to Host

To transmit a FIS to the host, the device shall proceed with the transmission in accordance with the Serial ATA 1.0 state machines. The device behavior is the same whether it is connected directly to the host or is connected to the host via a Port Multiplier.

When a device wants to transmit a FIS to the host, the Port Multiplier shall perform the following procedure:

1. After receiving an X_RDY_P primitive from the device, the Port Multiplier shall determine if the X bit is set in the device port's PSCR[1] (SError) register. The Port Multiplier shall not issue an R_RDY_P primitive to the device until the X bit is cleared to zero.

2. The Port Multiplier shall receive the FIS from the device. The Port Multiplier shall fill in the PM Port field with the port address of the transmitting device. The Port Multiplier shall transmit the modified FIS to the host with a recalculated CRC. The Port Multiplier shall check the CRC received from the device. If the CRC from the device is invalid the Port Multiplier shall corrupt the CRC sent to the host to ensure that the error condition is propagated. Refer to section 3.2.8.4 for specific details on how the Port Multiplier corrupts the CRC in this error case.

3. The Port Multiplier shall issue an X_RDY_P primitive to the host to start the transmission of the FIS to the host. After the host issues R_RDY_P to the Port Multiplier, the Port Multiplier shall transmit the FIS from the device to the host. The Port Multiplier shall not send R_OK_P status to the device until the host has issued an R_OK_P for the FIS reception. The R_OK_P status handshake shall be interlocked from the device to the host. Refer to section 3.2.1.

If an error is detected during any part of the FIS transfer, the Port Multiplier shall ensure that the error condition is propagated to the host and the device.

The Port Multiplier may wait for an X_RDY_P/R_RDY_P handshake with the host prior to issuing an R_RDY_P to the device to minimize buffering. The transfer between the device and Port Multiplier is handled separately from the transfer between the Port Multiplier and the host; only the end of frame R_OK_P handshake is interlocked. The Port Multiplier shall ensure that the flow control signaling latency requirement specified in section 7.4.7 of the Serial ATA 1.0 specification is met for all FIS transfers on a per link basis. Specifically, the Port Multiplier shall ensure that the flow control signaling latency is met between:

1. The host port and the host it is connected to
2. Each device port and the device that it is connected to

The Port Multiplier shall have Link and Phy layer state machines that comply with the Serial ATA 1.0 Link and Phy layer state machines.

3.2. Policies

3.2.1. FIS Delivery

The end of frame handshake shall be interlocked between the host and the device. Specifically, the Port Multiplier shall not issue an R_OK_P to the initiator of a FIS before the target of a FIS has issued an R_OK_P. The Port Multiplier shall propagate R_OK_P and R_ERR_P from the target of the FIS to the initiator of a FIS.

If a transmission fails before the R_OK_P handshake is delivered to the initiator, the Port Multiplier is responsible for propagating the error condition. Specifically, the Port Multiplier shall propagate $SYNC_P$ primitives received during a FIS transmission end-to-end to ensure that any error condition encountered in the middle of a FIS is propagated. Reference sections 3.2.8.4 and 3.2.8.5 on the appropriate actions to take when CRC calculation errors or possible data corruption occurs in a FIS transmission.

3.2.1.1. Port Priority

The Port Multiplier shall ensure that an enabled and active device port is not starved. The specific priority algorithm used is implementation specific.

The control port shall have priority over all device transfers. While a command is outstanding to the control port, no device transmissions shall be started by the Port Multiplier until the command outstanding to the control port is completed.

3.2.1.2. FIS Delivery Mechanisms (Informative)

This section provides an informative reference for one method that a Port Multiplier may use to satisfy the FIS Delivery policies outlined.

3.2.1.2.1. Starting a FIS Transmission (Informative)

If a device on a Port Multiplier asserts X_RDY_P and the Port Multiplier has selected that device for transmission next and the host port is not busy, the Port Multiplier will:

1. Issue X_RDY_P to the host.
2. Wait for the host to respond with R_RDY_P.
3. After the host issues R_RDY_P, the Port Multiplier will issue R_RDY_P to the device.

Then the transmission to the host may proceed.

If the host asserts X_RDY_P to the Port Multiplier and the Port Multiplier does not have X_RDY_P asserted to the host, the Port Multiplier responds with R_RDY_P to the host. The Port Multiplier receives the first Dword of the FIS payload from the host. If the Port Multiplier Port specified is a device port that is enabled on the Port Multiplier, the Port Multiplier issues an X_RDY_P over the device port specified and proceeds to transmit the entire FIS to the device. If a collision occurs during this process, the Port Multiplier will follow the procedures outlined in section 3.2.2.

3.2.1.2.2. Status Propagation (Informative)

If there is an on-going FIS transmission between the host and a device, the Port Multiplier only issues R_OK_P, R_ERR_P and $SYNC_P$ if it has first received that primitive from the host or device, unless a collision occurs or an invalid port is specified.

The Port Multiplier will not convey R_OK_P or R_ERR_P to the initiator of a FIS until the target of the FIS has issued R_OK_P or R_ERR_P once the end-to-end transmission has commenced.

If the initiator or target of a FIS transmission issues a $SYNC_P$ primitive during a FIS transfer, this primitive shall be propagated in order to ensure that the error condition is propagated to either end.

3.2.2. Collisions

A collision is when the Port Multiplier has already started a reception from the device that the host wants to transmit to. A collision also occurs when the device issues an X_RDY_P primitive at the same time that the Port Multiplier is issuing an X_RDY_P primitive to that device. All collisions are treated as an X_RDY_P/X_RDY_P collision; in accordance with Serial ATA 1.0, the host will lose all such collisions and must retransmit its FIS at a later time.

A collision only occurs when the host is trying to issue another Serial ATA native queued command to a device that has native queued commands outstanding. The Serial ATA native command queuing protocol guarantees that the host will never be transmitting a Data FIS when the collision occurs. This means that the Port Multiplier can safely issue an error to the host and that the host will retry the failed FIS transmission at a later point in time.

When the Port Multiplier detects a collision, the Port Multiplier shall finish reception of the FIS from the host and shall issue an R_ERR_P to the host for the end of frame handshake. The Port Multiplier shall discard the FIS received from the host. The Port Multiplier shall not return an R_RDY_P to the host until the Port Multiplier has transmitted the FIS from the affected device port to the host. This ensures that the Port Multiplier transmits the FIS from the device before another collision occurs due to the host retransmission. The host must attempt to retry the FIS transfer that failed.

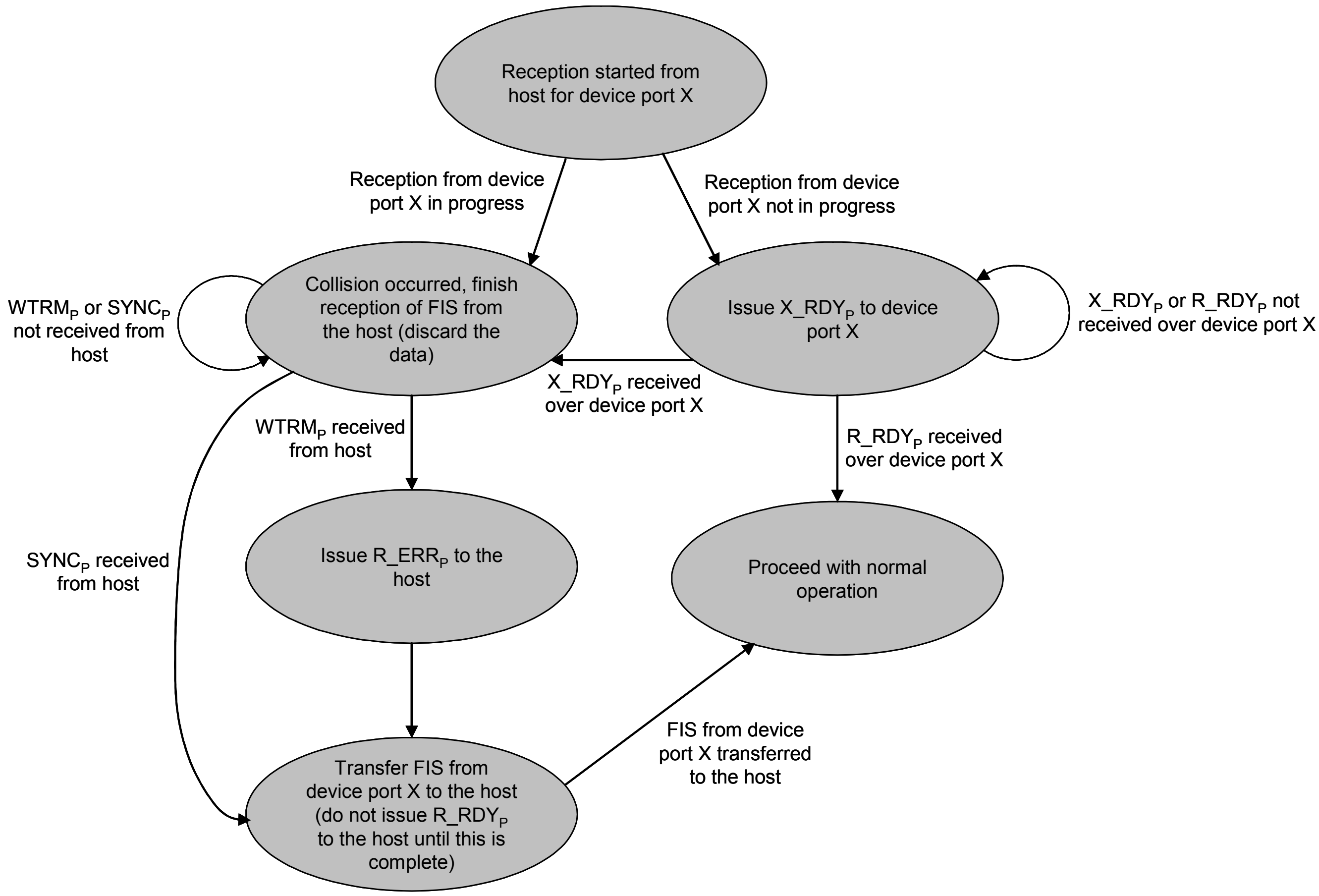

Figure 6 Collision Flow Graph

X_RDY_P/X_RDY_P collisions that occur on the host port shall be handled in accordance with the Link layer state diagrams in section 7.6 of the Serial ATA 1.0 specification.

3.2.3. Booting with Legacy Software

Booting is accommodated off of the first port of the Port Multiplier, device port 0h, without any special software or hardware support. An HBA that does not support attachment of a Port Multiplier will still work with the device on device port 0h.

If a system requires fast boot capability it should ensure that both the BIOS and OS driver software are Port Multiplier aware. If legacy software is used in the presence of a Port Multiplier and there is no device present on device port 0h, legacy software will detect a device as present but will never receive a Register FIS with the device signature. Waiting for the device signature may cause a legacy BIOS not to meet fast boot timing requirements.

3.2.4. Staggered Spin-up Support

The Port Multiplier shall disable all device ports on power-up or upon receiving a COMRESET signal over the host port. This feature allows the host to control when each device spins up. Refer to section 7.4.2 on how to enumerate a device, including devices that may be spun down because the port is disabled.

3.2.5. Hot Plug Events

Port Multiplier handling of hot plug events is defined by the hot plug state machines for the host port and device port, in sections 3.2.5.1 and section 3.2.5.2 respectively.

Upon receiving a COMRESET signal from the host, the Port Multiplier shall perform an internal reset. As part of the internal reset, the Port Multiplier shall update the Serial ATA superset Status and Control registers for each device port. A more detailed description of COMRESET handling is in section 7.2.1.

Upon receiving a COMINIT signal from a device, the Port Multiplier shall update the Serial ATA superset Status and Control registers for that port as defined in Serial ATA 1.0. The Port Multiplier shall set the X bit in the DIAG field of the device port's PSCR[1] (SError) register to mark that device presence has changed as defined in Serial ATA II: Extensions to Serial ATA 1.0. If the Port Multiplier has not received a FIS for the control port after the last COMRESET and the COMINIT signal was received over device port 0h, the Port Multiplier shall propagate the COMINIT signal to the host as specified in the hot plug state machine for the host port in section 3.2.5.1. For all other cases, the COMINIT signal shall not be propagated to the host.

If the X bit in the DIAG field of the PSCR[1] (SError) register is set for a device port, the Port Multiplier shall disallow FIS transfers with that port until the X bit has been cleared. When the Port Multiplier is disallowing FIS transfers with a device port, the Port Multiplier must ensure FISes the device is attempting to transmit are not dropped.

It is recommended that host software frequently query GSCR[32], defined in section 4.1.2, to determine if there has been a device presence change on a device port.

3.2.5.1. Hot Plug State Machine for Host Port

Cabled hot plug of the Port Multiplier host port shall be supported since Port Multipliers may not share the same power supply as the host. Therefore the Port Multiplier shall periodically poll for host presence by sending periodic COMINIT signals to the host after transitioning to the HPHP1:NoComm state.

The state machine enables device port 0 after communication has been established over the host port and also clears the X bit in the DIAG field of the PSCR[1] (SError) register for device port 0 after it is set when operating with legacy software. In addition, the state machine will propagate COMINIT signals received from device port 0 to the host when operating with legacy software. These accommodations allow device port 0 to work with legacy host software.

When operating with Port Multiplier aware software, the state machine treats device port 0 exactly like every other device port.

HPHP1: NoComm[1]	Set Port Multiplier to initial state as specified in section 7.1 and section 7.2.1		
	1. Unconditional	→	StartComm
	NOTE: 1. This state is entered upon power-on reset or in response to a received COMRESET		

HPHP2: StartComm	Transition Phy state machine to state DP1: DR_RESET. Communications retry interval reset to initial value.[1]		
	1. Unconditional	→	WaitComm
	NOTE: 1. The communications retry interval is vendor unique.		

HPHP3: WaitComm			
	1. PhyRdy asserted	→	EnablePort0
	2. PhyRdy not asserted and communications retry interval not expired	→	WaitComm
	3. PhyRdy not asserted and communications retry interval expired	→	NoComm

HPHP4: EnablePort0	Enable device port 0 if device port 0 is currently disabled, clear X bit in PSCR[1] (SError) for device port 0		
	1. PhyRdy asserted	→	LegacyCommOK
	2. PhyRdy not asserted	→	NoComm

HPHP5: LegacyCommOK			
	1. Communications lost and interface not in power management state (i.e. unplug) and FIS not received for control port	→	NoComm
	2. Communications established or interface in power management state and COMINIT not received on device port 0 and FIS not received for control port	→	LegacyCommOK
	3. Communications established or interface in power management state and COMINIT received on device port 0 and FIS not received for control port	→	StartComm[1]
	4. FIS received for control port	→	CommOK[2]
	NOTE: 1. The X bit in PSCR[1] (SError) for Port 0 shall be set prior to making the transition 2. If bit 0 of the DET field in PSCR[0] (SStatus) for Port 0 is set to one then the X bit in PSCR[1] (SError) for Port 0 shall be set prior to making the transition		

HPHP6: CommOK			
	1. Communications lost and interface not in power management state (i.e. unplug)	→	NoComm
	2. Communications established or interface in power management state	→	CommOK

3.2.5.2. Hot Plug State Machine for Device Port

The device port hot plug behavior is exactly the same as the behavior for a host controller supporting device hot plug directly. There is no change to the device and there is no change to the usage model.

DPHP1: NoComm			
	1. PhyRdy asserted	→	CommOK
	2. PhyRdy not asserted	→	NoComm

DPHP2: CommOK			
	1. Communications lost and interface not in power management state	→	NoComm
	2. Communications established or interface in power management state	→	CommOK

3.2.6. Link Power Management

In accordance with Serial ATA 1.0, the Port Multiplier must support reception of $PMREQ_P$ primitives from the host and from attached devices. If the Port Multiplier does not support the power management state requested, the Port Multiplier shall respond to the $PMREQ_P$ primitive with a $PMNAK_P$ primitive.

As described in the Serial ATA 1.0 specification for Link power management, before the Port Multiplier delivers a FIS, the Port Multiplier shall check the state of the link and issue a COMWAKE signal if the link is in partial or slumber. The Port Multiplier shall accurately reflect the current state of each device port link in the port specific registers (specifically PSCR[0] (SStatus) for each port) defined in section 4.2.

The Port Multiplier shall not propagate a $PMREQ_P$ primitive received on a device port. A $PMREQ_P$ primitive received from a device only affects the link between that device and the Port Multiplier. If the Port Multiplier receives a $PMREQ_P$ primitive over a device port, the Port Multiplier shall perform the following actions:

- The Port Multiplier shall respond to the device with $PMACK_P$ or $PMNAK_P$
- If the Port Multiplier responds to the device with $PMACK_P$:
  - The Port Multiplier shall transition the link with that device to the power state specified
  - The Port Multiplier shall update the port specific registers for that device port to reflect the current power management state of that link

The Port Multiplier shall propagate a $PMREQ_P$ primitive received from the host to all active device ports if the Port Multiplier responds to the $PMREQ_P$ primitive with $PMACK_P$. In this case, the Port Multiplier shall propagate the $PMREQ_P$ primitive to all device ports that have PhyRdy set. If a device responds to a $PMREQ_P$ primitive with $PMNAK_P$, this event shall only affect the link with that device. The host may interrogate the Port Multiplier port specific registers to determine which device ports are in a power managed state.

If the Port Multiplier receives a $PMREQ_P$ primitive from the host, the specific actions the Port Multiplier shall take are:

- The Port Multiplier shall respond to the host with $PMACK_P$ or $PMNAK_P$
- If the Port Multiplier responds to the host with $PMACK_P$:
  - The Port Multiplier shall transition the link with the host to the power state specified
  - The Port Multiplier shall issue $PMREQ_P$ primitives to all device ports that have PhyRdy set
    - If a device responds with $PMACK_P$, the Port Multiplier shall transition the link with that device to the power state specified and shall update the Port Multiplier port specific registers for that port
    - If a device responds with $PMNAK_P$, the Port Multiplier shall not take any action with that device port

The Port Multiplier shall wake device links on an as-needed basis. A COMWAKE signal from the host is not propagated to the devices. The Port Multiplier shall issue a COMWAKE signal to a device when a FIS needs to be delivered to that device.

The Port Multiplier may issue a $PMREQ_P$ to the host if all device ports are in partial/slumber or are disabled.

3.2.7. Reducing Context Switching Complexity

It may complicate some host controller designs if traffic from another device is received in the middle of certain FIS sequences. For example, if a host receives a DMA Activate FIS from one

device, it may be awkward for the host to receive a FIS from another device before it is able to issue the Data FIS to the first device.

The Port Multiplier shall provide the host with the opportunity to transmit before initiating any pending device transmissions. The Port Multiplier shall not assert an X_RDY_P primitive to the host until the Port Multiplier has received at least two consecutive $SYNC_P$ primitives from the host. In the case of a collision, the Port Multiplier shall ignore this requirement and perform the actions detailed in section 3.2.2.

If the host wants to transmit prior to receiving another FIS, the host should issue no more than one $SYNC_P$ primitive between the end of the last FIS transmission and the start of the next FIS transmission. One possible implementation is to transition to the host Link layer state HL_SendChkRdy defined in the Serial ATA 1.0 specification immediately, regardless of whether the host is ready to proceed with the next transmission. The host would then only transition out of HL_SendChkRdy state when the host is ready to proceed with the transmission and the R_RDY_P primitive is received.

3.2.8. Error Handling and Recovery

The host is responsible for handling error conditions in the same way it handles errors when connected directly to a device. The host is responsible for detecting commands that do not finish and performing error recovery procedures as needed. The exact host software error recovery procedures are implementation specific.

The Port Multiplier is not responsible for performing any error recovery procedures. The Port Multiplier shall return R_ERR_P for certain error conditions as described by the Serial ATA 1.0 state diagrams. The Port Multiplier shall update GSCR[32], defined in section 4.1.2, and PSCR[1] (SError) for the device port that experiences an error.

3.2.8.1. Command Timeout

If a command times out, the host may check PSCR[1] (SError), defined in section 4.2.2, to determine if there has been an interface error condition on the port that had the error. If there has been a device presence change on that port as indicated by the X bit in the DIAG field of PSCR[1] (SError) for the port, the host should re-enumerate the device on that port. The host software error recovery mechanism after a command timeout is implementation specific.

3.2.8.2. Disabled Device Port

If the host transmits a FIS to a device port that is disabled or that has FIS transfers disallowed due to the X bit being set in the DIAG field of PSCR[1] (SError) for that port, the Port Multiplier shall not perform an R_OK or R_ERR handshake at the end of FIS reception and shall instead terminate the FIS reception by issuing $SYNC_P$ primitives to the host. The host is responsible for detecting that the command did not finish and performing error recovery procedures, including clearing the X bit as necessary.

3.2.8.3. Invalid Device Port Address

An invalid device port address is a device port address that has a value greater than or equal to the number of device ports that the Port Multiplier supports. If the host specifies an invalid device port address as part of a FIS transmission, the Port Multiplier shall issue a $SYNC_P$ primitive and terminate reception of the FIS. The host is responsible for detecting that the command did not finish and performing normal error recovery procedures.

3.2.8.4. Invalid CRC for Device Initiated Transfer

On a device initiated transfer, the Port Multiplier must recalculate the CRC since it modifies the first Dword of the FIS. The Port Multiplier shall check the original CRC sent by the device. If the original CRC is invalid, the Port Multiplier shall invert the recalculated CRC to ensure that the CRC error is propagated to the host. The inversion can be done by XORing the recalculated

CRC with FFFFFFFFh. The Port Multiplier shall also update the error information in PSCR[1] (SError) for the device port that experienced the error.

3.2.8.5. Data corruption

If the Port Multiplier encounters an 8B/10B decoding error or any other error that could affect the integrity of the data passed between the initiator and target of a FIS, the Port Multiplier shall ensure that the error is propagated to the target. The Port Multiplier may propagate the error by corrupting the CRC for the FIS. The Port Multiplier shall also update the error information in PSCR[1] (SError) for the device port that experienced the error.

3.2.8.6. Unsupported Command Received on control port

If an unsupported command is received on the control port, the Port Multiplier shall respond with a Register FIS that has the values shown in Figure 7 for the Status and Error registers.

Register	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Error	Reserved (0)					ABRT	Reserved (0)	
Status	BSY	DRDY	DF	na	DRQ	0	0	ERR

Figure 7 Register values for an unsupported command

ABRT = 1
BSY = 0
DRDY = 1
DF = 0
DRQ = 0
ERR = 1

3.2.9. BIST Support

A Port Multiplier may optionally support BIST. A Port Multiplier that supports BIST shall only support BIST in a point-to-point manner. A Port Multiplier that supports BIST shall not propagate a BIST Activate FIS received on one port over another port. The host determines that a Port Multiplier supports BIST by checking GCSR[64], defined in section 4.1.3.

To enter BIST mode over the host connection with a Port Multiplier, the host shall issue a BIST Activate FIS to the Port Multiplier control port. The host shall not issue a BIST Activate FIS to a device port.

To enter BIST mode over a device connection, the device shall issue a BIST Activate FIS to the Port Multiplier. The Port Multiplier shall intercept the BIST Activate FIS and enter BIST mode. Upon entering BIST mode with the device, the Port Multiplier shall update the PSCR[0] (SStatus) register for that port to reflect that the link has entered BIST mode, as specified in section 10.1.1 of the Serial ATA 1.0 specification.

Initiation of BIST by a Port Multiplier is vendor unique.

4. Port Multiplier Registers

The Port Multiplier registers are accessed using Read/Write Port Multiplier commands issued to the control port.

4.1. General Status and Control Registers

The control port address is specified in the PortNum field of the Read/Write Port Multiplier command defined in section 5.1 and section 5.2 in order to read/write the General Status and Control registers.

4.1.1. Static Configuration Information

The Static Configuration Information section of the General Status and Control Registers contains registers that are static throughout the operation of the Port Multiplier. These registers are read-only and may not be modified by the host.

Register	O/M	F/V	Description
GSCR[0]	M		Product Identifier
		F	31-16 Device ID allocated by the vendor
		F	15-0 Vendor ID allocated by the PCI-SIG
GSCR[1]	M		Revision Information
		F	31-16 Reserved (0)
		F	15-8 Revision level of the Port Multiplier
		F	7-2 Reserved (0)
		F	1 1 = Supports Port Multiplier specification 1.0
		F	0 Reserved (0)
GSCR[2]	M		Port Information
		F	31-4 Reserved (0)
		F	3-0 Number of exposed device fan-out ports
GSCR[3] – GSCR[31]	M	F	Reserved (0)

Key:
O/M = Mandatory/optional requirement.
M = Support of the register is mandatory.
O = Support of the register is optional.
F/V = Fixed/variable content
F = the content of the register is fixed and does not change.
V = the contents of the register is variable and may change.
X = the content of the register may be fixed or variable.

Figure 8 Static Information Registers

Register 0: Product Identifier

Support for this register is mandatory. The register identifies the vendor that produced the Port Multiplier and the specific device identifier.

Bit 15-0 shall be set to the vendor identifier allocated by the PCI-SIG of the vendor that produced the Port Multiplier.

Bit 31-16 shall be set to a device identifier allocated by the vendor.

Register 1: Revision Information

Support for this register is mandatory. The register specifies the specification revision that the Port Multiplier supports. The register also specifies the revision level of the specific Port Multiplier product identified by Register 0.

Bit 0 is reserved and shall be cleared to zero.

Bit 1 when set to 1 indicates that the Port Multiplier supports Port Multiplier specification version 1.0

Bit 7-2 are reserved and shall be cleared to zero.

Bit 15-8 identifies the revision level of the Port Multiplier product identified by Register 0. This identifier is allocated by the vendor.

Bit 31-16 are reserved and shall be cleared to zero.

Register 2: Port Information

Support for this register is mandatory. The register specifies information about the ports that the Port Multiplier contains, including the number of exposed device fan-out ports. The number of exposed device ports is the number of device ports that are physically connected and available for use on the product.

Bit 3-0 specifies the number of exposed device fan-out ports. A value of zero is invalid. The control port shall not be counted.

Bit 31-4 are reserved and shall be cleared to zero.

Register 3-31: Reserved

Registers 3-31 are reserved for future Port Multiplier definition and shall be cleared to zero.

4.1.2. Status Information and Control

The Status Information section of the General Status and Control Registers contains registers that convey status information and control operation of the Port Multiplier.

Register	O/M	F/V	Description		
GSCR[32]	M		Error Information		
		F		31-15	Reserved (0)
		V		14	OR of selectable bits in Port 14 PSCR[1] (SError)
		V		13	OR of selectable bits in Port 13 PSCR[1] (SError)
		V		12	OR of selectable bits in Port 12 PSCR[1] (SError)
		V		11	OR of selectable bits in Port 11 PSCR[1] (SError)
		V		10	OR of selectable bits in Port 10 PSCR[1] (SError)
		V		9	OR of selectable bits in Port 9 PSCR[1] (SError)
		V		8	OR of selectable bits in Port 8 PSCR[1] (SError)
		V		7	OR of selectable bits in Port 7 PSCR[1] (SError)
		V		6	OR of selectable bits in Port 6 PSCR[1] (SError)
		V		5	OR of selectable bits in Port 5 PSCR[1] (SError)
		V		4	OR of selectable bits in Port 4 PSCR[1] (SError)
		V		3	OR of selectable bits in Port 3 PSCR[1] (SError)
		V		2	OR of selectable bits in Port 2 PSCR[1] (SError)
		V		1	OR of selectable bits in Port 1 PSCR[1] (SError)
		V		0	OR of selectable bits in Port 0 PSCR[1] (SError)
GSCR[33]	M		Error Information Bit Enable		
		V		31-0	If set, bit is enabled for use in GSCR[32]
GSCR[34] – GSCR[63]	M	F	Reserved (0)		

Key:
O/M = Mandatory/optional requirement.
M = Support of the register is mandatory.
O = Support of the register is optional.
F/V = Fixed/variable content
F = the content of the register is fixed and does not change.
V = the contents of the register is variable and may change.
X = the content of the register may be fixed or variable.

Figure 9 Status Information and Control Registers

Register 32: Error Information

Support for this register is mandatory. This register reflects whether specific bits have been set in any of the device port's PSCR[1] (SError) register. A set bit may reflect an error or that device presence has changed. The host selects the device port's PSCR[1] (SError) bits to reflect using GSCR[33]. The Port Multiplier algorithm for updating the register contents is:

```
for (n=0; n<NumPorts; n++)
{
      if (Port[n].PSCR[1] & GSCR[33]) == 0)
            GSCR[32].Bit[n]=0
      else
            GSCR[32].Bit[n]=1
}
```

This register is read-only. The specific device fan-out port's PSCR[1] (SError) register must be written in order to clear values in the affected PSCR[1] (SError) register and by reflection in this register. Refer to section 10.1.2 in the Serial ATA 1.0 specification for the definition of the SError register.

Bit 0 shall be set to the logically OR-ed value of selected bits in Port 0 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33].

Bit 1 shall be set to the logically OR-ed value of selected bits in Port 1 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 1 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 2 shall be set to the logically OR-ed value of selected bits in Port 2 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 2 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 3 shall be set to the logically OR-ed value of selected bits in Port 3 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 3 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 4 shall be set to the logically OR-ed value of selected bits in Port 4 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 4 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 5 shall be set to the logically OR-ed value of selected bits in Port 5 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 5 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 6 shall be set to the logically OR-ed value of selected bits in Port 6 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 6 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 7 shall be set to the logically OR-ed value of selected bits in Port 7 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 7 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 8 shall be set to the logically OR-ed value of selected bits in Port 8 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 8 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 9 shall be set to the logically OR-ed value of selected bits in Port 9 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 9 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 10 shall be set to the logically OR-ed value of selected bits in Port 10 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 10 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 11 shall be set to the logically OR-ed value of selected bits in Port 11 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 11 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 12 shall be set to the logically OR-ed value of selected bits in Port 12 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 12 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 13 shall be set to the logically OR-ed value of selected bits in Port 13 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 13 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 14 shall be set to the logically OR-ed value of selected bits in Port 14 PSCR[1] (SError). The bits to be OR-ed are selected using GSCR[33]. If Port 14 is not implemented by the Port Multiplier, this bit shall be cleared to zero.

Bit 31-15 is reserved and shall be cleared to zero.

Register 33: Error Information Bit Enable

Support for this register is mandatory. This register selects/enables bits to be used for the OR operation in GSCR[32]. If a bit is set to one, that bit will be reflected for each device port in GSCR[32]. This is a global enable and is not device port specific. The default value of this register shall be 0400FFFFh; that corresponds to all bits in the ERR field and the DIAG field’s X bit being OR-ed together for each device port in GSCR[32].

Register 34-63: Reserved

Registers 34-63 are reserved for future Port Multiplier definition and shall be cleared to zero.

4.1.3. Features Supported

The Features Supported section of the General Status and Control Registers contains registers that convey the optional features that are supported by the Port Multiplier. The Features Supported registers have a one-to-one correspondence with the Features Enabled registers defined in section 4.1.4. All Features Supported registers are read-only.

Register	O/M	F/V	Description
GSCR[64]	M	 F F F F	Port Multiplier Revision 1.0 Optional Features Support 31-3 Reserved (0) 2 1 = Supports dynamic SSC transmit enable 1 1 = Supports issuing $PMREQ_P$ to host 0 1 = Supports BIST
GSCR[65] – GSCR[95]	M	F	Reserved (0)
Key: O/M = Mandatory/optional requirement. M = Support of the register is mandatory. O = Support of the register is optional. F/V = Fixed/variable content F = the content of the register is fixed and does not change. V = the contents of the register is variable and may change. X = the content of the register may be fixed or variable.			

Figure 10 Features Supported Registers

Register 64: Port Multiplier Revision 1.0 Optional Features Support

Support for this register is mandatory. This register specifies the optional Port Multiplier specification revision 1.0 features that the Port Multiplier supports.

Bit 0 if set to 1 indicates that the Port Multiplier supports BIST as outlined in section 3.2.9. If this bit is set to 0 the Port Multiplier does not support reception of the BIST Activate FIS.

Bit 1 if set to 1 indicates that the Port Multiplier supports issuing $PMREQ_P$ requests to the host when all device ports are disabled or in a partial/slumber state. If this bit is set to 0, the Port Multiplier does not support issuing $PMREQ_P$ requests to the host.

Bit 2 if set to 1 indicates that the Port Multiplier supports dynamically enabling and disabling spread-spectrum clocking transmit. If this bit is set to 0, the Port Multiplier does not support dynamically enabling and disabling spread-spectrum clocking transmit.

Bit 31-3 are reserved and shall be cleared to zero.

Register 65-95: Reserved

Registers 65-95 are reserved for future Port Multiplier definition and shall be cleared to zero.

4.1.4. Features Enabled

The Features Enabled section of the General Status and Control Registers contains registers that allow optional features to be enabled. The Features Enabled registers have a one-to-one correspondence with the Features Supported registers defined in section 4.1.3. All Features Enabled registers are read/write.

Register	O/M	F/V	Description
GSCR[96]	M	 F V V V	Port Multiplier Revision 1.0 Optional Features Enable 31-3 Reserved (0) 2 1 = Dynamic SSC transmit is enabled 1 1 = Issuing $PMREQ_P$ to host is enabled 0 1 = BIST support is enabled
GSCR[97] – GSCR[127]	M	F	Reserved (0)
Key: O/M = Mandatory/optional requirement. M = Support of the register is mandatory. O = Support of the register is optional. F/V = Fixed/variable content F = the content of the register is fixed and does not change. V = the contents of the register is variable and may change. X = the content of the register may be fixed or variable.			

Figure 11 Features Enabled Registers

Register 96: Port Multiplier Revision 1.0 Optional Features Enable

Support for this register is mandatory. This register controls whether the optional Port Multiplier specification revision 1.0 features that the Port Multiplier supports are enabled.

Bit 0 if set to 1 indicates that the Port Multiplier supports BIST and that BIST support is enabled. If this bit is set to 0 reception of the BIST Activate FIS is not enabled. If the Port Multiplier supports BIST, this bit shall be set to 1 on power-up or after a reset, otherwise it shall be cleared to 0 on power-up and reset.

Bit 1 if set to 1 indicates that the Port Multiplier supports issuing $PMREQ_P$ requests to the host when all device ports are disabled or in a partial/slumber state and the feature is enabled. If this bit is set to 0, the Port Multiplier shall not issue $PMREQ_P$ requests to the host. This bit shall be set to 0 on power-up and reset.

Bit 2 if set to 1 indicates that the Port Multiplier supports dynamically enabling and disabling spread-spectrum clocking transmit and that the feature is currently enabled. If this bit is set to 0 and the feature is supported as specified in GSCR[64], the Port Multiplier shall not use spread-spectrum clocking transmit. If this feature is supported as specified in GSCR[64], the bit shall be set to 1 on power-up and reset. If this feature is not supported as specified in GSCR[64], then the bit shall be set to 0 on power-up and reset.

Bit 31-3 are reserved and shall be cleared to zero.

Register 97-127: Reserved

Registers 97-127 are reserved for future Port Multiplier definition and shall be cleared to zero.

4.1.5. Vendor Unique

The Vendor Unique section of the General Status and Control Registers contains registers that are vendor unique.

<table>
<tr><th>Register</th><th>O/M</th><th>F/V</th><th>Description</th></tr>
<tr><td>GSCR[128] –
GSCR[255]</td><td>O</td><td>X</td><td>Vendor Unique</td></tr>
<tr><td colspan="4">Key:
O/M = Mandatory/optional requirement.
M = Support of the register is mandatory.
O = Support of the register is optional.
F/V = Fixed/variable content
F = the content of the register is fixed and does not change.
V = the contents of the register is variable and may change.
X = the content of the register may be fixed or variable.</td></tr>
</table>

Figure 12 Vendor Unique Registers

Register 128-255: Vendor unique

4.2. Port Status and Control Registers

The Port Multiplier shall maintain a set of Port Status and Control registers (PSCRs) for each device port that it supports. These registers are port specific and contain the Serial ATA superset Status and Control registers for each port. The host specifies the device port to read or write registers for in the PortNum field of the Read/Write Port Multiplier command defined in section 5.1 and section 5.2. The registers are defined in the following table.

PSCR[0]	SStatus register
PSCR[1]	SError register
PSCR[2]	SControl register
PSCR[3]	SActive register (not implemented)
PSCR[4]	Reserved
…	…
PSCR[15]	Reserved

Figure 13 PSCR definition

4.2.1. PSCR[0] – SStatus register

The PSCR[0] (SStatus) register is the 32-bit Serial ATA superset SStatus register defined in the Serial ATA 1.0 specification, section 10.1.1 and further defined in the Serial ATA II: Extensions to Serial ATA 1.0 specification section 5.1.3.

PSCR0	Serial ATA SStatus Register

Figure 14 PSCR[0] (SStatus) register definition

4.2.2. PSCR[1] – SError register

The PSCR[1] (SError) register is the 32-bit Serial ATA superset SError register defined in the Serial ATA 1.0 specification, section 10.1.2 and further defined in the Serial ATA II: Extensions to Serial ATA 1.0 specification section 5.1.2.

PSCR1	Serial ATA SError Register

Figure 15 PSCR[1] (SError) register definition

4.2.3. PSCR[2] – SControl register

The PSCR[2] (SControl) register is the 32-bit Serial ATA superset SControl register defined in the Serial ATA 1.0 specification, section 10.1.3 and further defined in the Serial ATA II: Extensions to Serial ATA 1.0 specification section 5.1.4. The reset value for the DET field shall be 4h.

PSCR2	Serial ATA SControl Register

Figure 16 PSCR[2] (SControl) register definition

5. Port Multiplier Command Definitions

5.1. Read Port Multiplier

The Read Port Multiplier command is used to read a register on a Port Multiplier. The Read Port Multiplier command must be issued to the control port.

5.1.1. Inputs

Register	7	6	5	4	3	2	1	0
Features	RegNum							
Features (exp)	na							
Sector Count	Reserved (0)							
Sector Count (exp)	na							
Sector Number	Reserved (0)							
Sector Number (exp)	na							
Cylinder Low	Reserved (0)							
Cylinder Low (exp)	na							
Cylinder High	Reserved (0)							
Cylinder High (exp)	na							
Device/Head	na				PortNum			
Command	E4h							

Figure 17 Read Port Multiplier command definition

PortNum	Set to the port address that has register to be read
RegNum	Set to number of register to read
na	Field or register is not used

5.1.2. Success Outputs

Upon successful completion, the ERR bit in the Status register shall be cleared and the value in the Error register shall be zero, and the value of the specified register shall be returned.

Register	7	6	5	4	3	2	1	0
Error	0							
Sector Count	Value (7:0)							
Sector Count (exp)	na							
Sector Number	Value (15:8)							
Sector Number (exp)	na							
Cylinder Low	Value (23:16)							
Cylinder Low (exp)	na							
Cylinder High	Value (31:24)							
Cylinder High (exp)	na							
Device/Head	Reserved (0)							
Status	BSY	DRDY	DF	na	DRQ	0	0	ERR

Figure 18 Read Port Multiplier success status result values

Value Set to the 32-bit value read from the register
na Field or register is not used

BSY = 0
DRDY = 1
DF = 0
DRQ = 0
ERR = 0

5.1.3. Error Outputs

Upon encountering an error, the Port Multiplier shall set the ERR bit in the Status register and identify the error code in the Error register.

Register	7	6	5	4	3	2	1	0
Error	Reserved (0)					ABRT	REG	PORT
Sector Count	Reserved (0)							
Sector Count (exp)	na							
Sector Number	Reserved (0)							
Sector Number (exp)	na							
Cylinder Low	Reserved (0)							
Cylinder Low (exp)	na							
Cylinder High	Reserved (0)							
Cylinder High (exp)	na							
Device/Head	Reserved (0)							
Status	BSY	DRDY	DF	na	DRQ	0	0	ERR

Figure 19 Read Port Multiplier error status result values

ABRT = 0
REG Set to one if the register specified is invalid
PORT Set to one if the port specified is invalid
na Field or register is not used

BSY = 0
DRDY = 1
DF = 0
DRQ = 0
ERR = 1

5.2. Write Port Multiplier

The Write Port Multiplier command is used to write a register on a Port Multiplier. The Write Port Multiplier command must be issued to the control port.

5.2.1. Inputs

Register	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Features	RegNum							
Features (exp)	na							
Sector Count	Value (7:0)							
Sector Count (exp)	na							
Sector Number	Value (15:8)							
Sector Number (exp)	na							
Cylinder Low	Value (23:16)							
Cylinder Low (exp)	na							
Cylinder High	Value (31:24)							
Cylinder High (exp)	na							
Device/Head	na				PortNum			
Command	E8h							

Figure 20 Write Port Multiplier command definition

Value Set to the 32-bit value to write to the register
RegNum Set to number of register to write
PortNum Set to the port address that has register to be written
na Field or register is not used

5.2.2. Success Outputs

Upon successful completion, the Port Multiplier shall write the specified value to the specified register. The ERR bit in the Status register shall be cleared and the value in the Error register shall be zero.

Register	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Error	0							
Sector Count	Reserved (0)							
Sector Count (exp)	na							
Sector Number	Reserved (0)							
Sector Number (exp)	na							
Cylinder Low	Reserved (0)							
Cylinder Low (exp)	na							
Cylinder High	Reserved (0)							
Cylinder High (exp)	na							
Device/Head	Reserved (0)							
Status	BSY	DRDY	DF	na	DRQ	0	0	ERR

Figure 21 Write Port Multiplier success status result values

na Field or register is not used

BSY = 0
DRDY = 1
DF = 0

DRQ = 0
ERR = 0

5.2.3. Error Outputs

Upon encountering an error, the Port Multiplier shall set the ERR bit in the Status register and identify the error code in the Error register.

Register	7	6	5	4	3	2	1	0
Error	Reserved (0)					ABRT	REG	PORT
Sector Count	Reserved (0)							
Sector Count (exp)	na							
Sector Number	Reserved (0)							
Sector Number (exp)	na							
Cylinder Low	Reserved (0)							
Cylinder Low (exp)	na							
Cylinder High	Reserved (0)							
Cylinder High (exp)	na							
Device/Head	Reserved (0)							
Status	BSY	DRDY	DF	na	DRQ	0	0	ERR

Figure 22 Write Port Multiplier error status result values

ABRT = 0
REG Set to one if the register specified is invalid
PORT Set to one if the port specified is invalid
na Field or register is not used

BSY = 0
DRDY = 1
DF = 0
DRQ = 0
ERR = 1

5.3. Interrupts

The Port Multiplier shall generate an interrupt in the status response to a command for the control port by setting the I bit in the Register FIS. The Port Multiplier shall not generate an interrupt in response to a software reset for the control port.

6. Serial ATA Superset Registers Enhancements

The host controller must provide a means by which software can set the PM Port field in all transmitted FISes. This capability may be exposed to software by minor enhancements to the SControl register detailed in section 6.1. In addition a means by which software can cause a specific device port to transition to a low power management state must be provided. This capability is exposed to software by adding an additional field called SPM to the SControl register as detailed in section 6.1.

6.1. SControl Register Enhancements

The Serial ATA interface control register (SControl) as defined in the Serial ATA 1.0 specification, section 10.1.3, provides a means by which software controls Serial ATA interface capabilities. In order to set the PM Port field, an additional field called PMP is added to the SControl register. A field called SPM has also been added to allow the host to initiate power management states.

SCR2	Reserved (0)	PMP	SPM	IPM	SPD	DET

Figure 23 Enhancement to SControl register

DET As defined in Serial ATA 1.0

SPD As defined in Serial ATA 1.0

IPM As defined in Serial ATA 1.0

SPM The Select Power Management (SPM) field is used to select a power management state. A non-zero value written to this field shall cause the power management state specified to be initiated. A value written to this field is treated as a one-shot. This field shall be read as ‘0000’.

0000 No power management state transition requested
0001 Transition to the PARTIAL power management state initiated
0010 Transition to the SLUMBER power management state initiated
0100 Transition to the active power management state initiated
All other values reserved

PMP The Port Multiplier Port (PMP) field represents the 4-bit value to be placed in the PM Port field of all transmitted FISes. This field is ‘0000’ upon power-up. This field is optional and an HBA implementation may choose to ignore this field if the FIS to be transmitted is constructed via an alternative method.

Reserved All reserved fields shall be cleared to zero.

7. Resets and Software Initialization Sequences

7.1. Power-up

On power-up, the Port Multiplier shall enter state HPHP1:NoComm in the hot plug state machine for the host port, specified in section 3.2.5.1. Upon entering this state, the Port Multiplier shall:

1. Clear any internal state and reset all parts of the Port Multiplier hardware.
2. Place the reset values in all Port Multiplier registers, including port specific registers. The reset values shall disable all device ports.

After performing this sequence, the Port Multiplier shall proceed with the actions described in the hot plug state machine for the host port.

If the Port Multiplier receives a COMRESET from the host before issuing the initial COMINIT signal to the host, the Port Multiplier shall immediately perform the actions detailed in section 7.2.1 and cease performing the power-up sequence listed above.

7.2. Resets

In Serial ATA 1.0, there are three mechanisms to reset a device: COMRESET, software reset, and the DEVICE RESET command for PACKET devices.

To reset a Port Multiplier, the host must issue a COMRESET. A Port Multiplier does not reset in response to a software reset or a DEVICE RESET command. The specific actions that a Port Multiplier takes in response to each reset type is detailed in the following sections.

7.2.1. COMRESET

When the Port Multiplier receives a COMRESET over the host port, the Port Multiplier shall enter state HPHP1:NoComm in the hot plug state machine for the host port, specified in section 3.2.5.1. Upon entering this state, the Port Multiplier shall:

1. Clear any internal state and reset all parts of the Port Multiplier hardware.
2. Place the reset values in all Port Multiplier registers, including port specific registers. The reset values shall disable all device ports.

After performing this sequence, the Port Multiplier shall proceed with the actions described in the hot plug state machine for the host port.

There is no functional difference between COMRESET and Power-On Reset.

7.2.2. Software Reset

When the host issues a software reset to the control port, two Registers FISes will be sent to the control port as a result. In the first Register FIS, the SRST bit in the Device Control register will be asserted. In the second Register FIS, the SRST bit in the Device Control register will be cleared.

Upon receiving the Register FIS with the SRST bit asserted, the Port Multiplier shall wait for the Register FIS that has the SRST bit cleared before issuing a Register FIS with the Port Multiplier signature to the host. The Port Multiplier's behavior shall be consistent with the Software reset protocol described in section 9.3 of the Serial ATA 1.0 specification. The values to be placed in the Register FIS are listed in Figure 24.

Register	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Error	00h							
Sector Count	01h							
Sector Count (exp)	00h							
Sector Number	01h							
Sector Number (exp)	00h							
Cylinder Low	69h							
Cylinder Low (exp)	00h							
Cylinder High	96h							
Cylinder High (exp)	00h							
Device/Head	00h							
Status	BSY	DRDY	DF	na	DRQ	0	0	ERR

Figure 24 Software reset to control port result values

BSY = 0
DRDY = 1
DF = 0
DRQ = 0
ERR = 0

The Port Multiplier shall take no reset actions based on the reception of a software reset. The only action that a Port Multiplier shall take is to respond with a Register FIS that includes the Port Multiplier signature. To cause a general Port Multiplier reset, the COMRESET mechanism is used.

7.2.3. Device Reset

A device reset command issued to the control port shall be treated as an unsupported command by the Port Multiplier. Refer to section 3.2.8.6.

7.3. Software Initialization Sequences (Informative)

This section details the sequences that host software should take to initialize a Port Multiplier device.

7.3.1. Port Multiplier Aware Software (Informative)

Port Multiplier aware software will check the host's SStatus register to determine if a device is connected to the port. If a device is connected to the port, the host will issue a software reset to the control port. If the Port Multiplier signature is returned, then a Port Multiplier is attached to the port. Then the host will proceed with enumeration of devices on Port Multiplier ports as detailed in section 7.4.2.

Port Multiplier aware software shall not require a device to be present on device port 0h in order to determine a Port Multiplier is present.

7.3.2. Legacy Software (Informative)

Legacy software will wait for the signature of the attached device to be returned to the host. If a device is present on device port 0h, the device connected to device port 0h will return a Register FIS to the host that contains its signature. If a device is not present on device port 0h, the legacy software will timeout waiting for the signature to be returned and will assume that a device failure has occurred. If fast boot is a requirement, the system should have a Port Multiplier aware BIOS and Port Multiplier aware OS driver.

When legacy software is loaded, all device ports other than device port 0h are disabled. The host will only receive FISes from the device attached to device port 0h.

7.3.3. Boot Devices Connected to Port Multiplier (Informative)

System designers should only connect multiple boot devices to a Port Multiplier if the BIOS is Port Multiplier aware. For example, in a system that contains three bootable devices (hard drive, CD-ROM, and DVD) these devices should only be attached to the Port Multiplier if the BIOS is Port Multiplier aware or the user will not be able to boot off of the devices that are not connected to device port 0.

7.4. Port Multiplier Discovery and Device Enumeration (Informative)

7.4.1. Port Multiplier Discovery (Informative)

To determine if a Port Multiplier is present, the host performs the following procedure. The host will determine if communication is established on the host's Serial ATA port by checking the host's SStatus register. If a device is present, the host will issue a software reset with the PM Port field set to the control port. The host will check the signature value returned and if it corresponds to the Port Multiplier Signature, the host knows that a Port Multiplier is present. If the signature value does not correspond to a Port Multiplier, the host may proceed with the normal initialization sequence for that device type. The host shall not rely on a device being attached to device port 0h to determine that a Port Multiplier is present.

When a Port Multiplier receives a software reset to the control port, the Port Multiplier shall issue a Register FIS to the host according to the procedure detailed in section 7.2.2. The signature value contained in the Register FIS is shown in Figure 25 .

	Sector Count	Sector Number	Cylinder Low	Cylinder High	Device
Port Multiplier Signature	01h	01h	69h	96h	00h

Figure 25 Port Multiplier Signature

7.4.2. Device Enumeration (Informative)

After discovering a Port Multiplier, the host will enumerate all devices connected to the Port Multiplier. The host will read GSCR[2], defined in section 4.1.1, to determine the number of device ports on the Port Multiplier.

For each device port on the Port Multiplier, the host performs the following procedure to enumerate a device connected to that port:

1. The host will enable the device port. The host can enable a device port by setting the DET field appropriately in the device port's PSCR[2] (SControl) register, as specified in section 10.1.3 of the Serial ATA 1.0 specification. The host uses the Write Port Multiplier command to write PSCR[2] (SControl) for the device port to be enabled.

2. The host should allow for communication to be established and device presence to be detected after enabling a device port. Refer to section 6.8.1 of the Serial ATA 1.0 specification that describes the host PHY initialization sequence.

3. The host reads PSCR[0] (SStatus) for the device port using the Read Port Multiplier command. If PSCR[0] (SStatus) indicates that a device is present, the host queries PSCR[1] (SError) for the device port and clears the X bit indicating device presence has changed.

4. The signature generated by the device as a consequence of the initial COMRESET to the device port may be discarded if the host does not support context switching because the BSY bit may be clear when the Register FIS is received by the host. Therefore, to determine the signature of the attached device, the host should issue a software reset to the device port. If a valid signature is returned for a recognized device, the host may then proceed with normal initialization for that device type.

Cascading Port Multipliers shall not be supported.

APPENDIX A. SWITCHING TYPES (INFORMATIVE)

The host may use two different switching types depending on the capabilities of the host controller. If the host controller supports hardware context switching based on the value of the PM Port field in a received FIS, then the host may have commands outstanding to multiple devices at the same time. This switching type is called FIS-based switching and results in FISes being delivered to the host from any of the devices that have commands outstanding to them. If the host controller does not support hardware context switching based on the value of the PM Port field in a received FIS, then the host may only have commands outstanding to one device at any point in time. This switching type is called command-based switching and results in FISes being delivered to the host from only the one device that has commands outstanding to it. The Port Multiplier's operation is the same regardless of the switching type used by the host.

A.1 Command-based switching (Informative)

Host controllers that do not support hardware context switching utilize a switching type called command-based switching. To use command-based switching, the host controller has commands outstanding to only one device at any point in time. By only issuing commands to one device at a time, the result is that the Port Multiplier only delivers FISes from that device.

A host controller may support command-based switching by implementing the Port Multiplier Port (PMP) field in the SControl register as detailed in section 6.1. In order to use this mechanism, host software would set the PMP field appropriately before issuing a command to a device connected to the Port Multiplier. When host software had completed the commands with a particular device port, it would modify the PMP field before issuing commands to any other device port. Note that the PMP field must be set to the control port when host software would like to issue commands to the Port Multiplier itself (such as Read Port Multiplier or Write Port Multiplier).

A.2 FIS-based switching (Informative)

Host controllers that support hardware context switching may utilize a switching type called FIS-based switching. FIS-based switching allows the host controller to have commands outstanding to multiple devices at any point in time. When commands are outstanding to multiple devices, the Port Multiplier may deliver FISes from any device with commands outstanding.

A.2.1 Host Controller Requirements (Informative)

To support FIS-based switching, the host controller must have context switching support. The host controller must provide a means for exposing a programming interface for up to 16 devices on a single port. The context switching support must comprehend not only Control Block register context, but also DMA engine context and the SActive register. The host controller must be able to update context for a particular device even if the programming interface for that device is not currently selected by host software.

The host controller must fill in the PM Port field in hardware assisted FIS transmissions. For example, if the host controller receives a DMA Activate from a device, it must be able to construct a Data FIS with the PM Port field set to the value in the received DMA Activate FIS.

Refer to section 3.2.7 for a mechanism to reduce the complexity of host context switching.